AMENDMENTS TO THE CLAIMS

This listing of the claims will replace all prior versions, and listings, of claims in the application:

1. (Currently Amended) A method of generating an intermediate representation of a register-based program code written for running on a programmable machine having a set of registers, ~~including~~wherein the set of registers include at least one variable size register which is addressable by ~~said~~the program code in a plurality of different widths, ~~said~~the method comprising the computer-implemented steps of:

~~(i)~~ ——generating a plurality of register objects each representing a respective one of ~~said~~the registers as referenced by the program ~~codefor holding variable values to be generated by the program code~~, wherein ~~said~~the at least one variable ~~sized~~size register is represented by an associated set of ~~said~~the register objects ~~with~~having one of ~~said~~the register objects ~~being provided~~ for each different width of the variable ~~sized~~size register; ~~and~~

~~(ii)~~ ——generating a plurality of expression objects each representing ~~fixed values and/or relationships between said fixed values and said variable values~~a respective operator or operand ~~according to~~in ~~said~~the program code relating to ~~said~~the registers; and

forming a network of ~~said~~the register objects and the expression objects, wherein ~~with~~ each of ~~said~~the expression objects ~~being~~is referenced by one or more of ~~said~~the register objects to which it relates.

~~——wherein at least one variable sized register is represented by plural register objects, one register object being provided for each possible size of the variable sized register.~~

2. (Currently Amended) ~~A~~The method according to claim 1, wherein a write operating in ~~said~~the program code to ~~a said~~the variab~~l~~ye sized register at a particular width is ~~effected~~ represented in the intermediate representation by writing to the register object corresponding to the appropriate ~~size~~ width and maintaining a record of which of the register objects contain valid data.

3. (Currently Amended) ~~A~~The method according to claim 2, wherein a read operation from a variable~~y~~ size~~d~~ register is ~~effected~~ represented in the intermediate representation by determining from ~~said~~the record if there is valid data in more than one corresponding register object which must be combined to give the same effect as reading from the variable~~y~~ size~~d~~ register, and

(i) if it is determined that no such combination is required, reading from the appropriate register object; and

(ii) if it is determined that such combination is required, combining the contents of appropriate register objects to provide a real value.

4. (Currently Amended) The method of claim 1, further comprising translating the program code written for execution by a processor of a first type so that the program code may be executed by a processor of a second type, using the generated intermediate representation.

5. (Currently Amended) The method of claim 4, wherein the ~~translation~~ translating step is performed dynamically as the program code is run.

6. (Currently Amended) The method of claim 1, further comprising optimizing the program code by optimizing ~~said~~the generated intermediate representation.

7. (Currently Amended) The method of claim 6, wherein the optimizing step is used to optimize the program code written for execution by a processor of a first type so that the program code may be executed more efficiently by that processor.

8. (Currently Amended) A method of generating an intermediate representation of program code expressed in terms of ~~the~~ an instruction set of a subject processor comprising at least one variable size~~d~~ register which is accessible by the program code in a plurality of different field widths, the method comprising the computer implemented steps of:

generating a set of associated ~~abstract~~ register objects each representing a different field width of the variable size~~d~~ register;

for each write operation of a certain field width to the variable size~~d~~ register, writing to ~~an abstract~~ the register object of the same width;

maintaining a record of which ~~abstract~~ of ~~said~~the register objects contain valid data, which record is updated upon each write operation; and

for each read operation of a given field width, determining from ~~said~~the record whether there is valid data in more than one of ~~said~~the ~~different sized abstract~~ register~~s~~ objects of the set which must be combined to give the same effect as the same read operation performed upon the variable size register; and

(a) if it is determined that no combination is so required, reading directly from the appropriate register object; or

(b) if it is determined that data from more than one register object must be so combined, combining the contents of those registers objects.

9. (Currently Amended) The method according to claim 48, wherein the step of determining whether or not the contents of more than one ~~abstract~~ register object must be combined and if so which ~~abstract~~ registers objects must be combined, is determined in accordance with the following conditions in respect of ~~each~~ ~~said~~the set of ~~different sized abstract registers~~associated register objects:

(i) if ~~the~~ data required for an access to the variable ~~sized~~size register lies wholly within ~~one~~ a valid one of ~~said~~the ~~abstract~~ register objects, the register object only is accessed; and

(ii) if the data required for an access lies within more than one of ~~said~~the ~~valid abstract register~~register objects, data is combined from ~~said~~the more than one register objects ~~those valid abstract registers~~ to perform the access.

10. (Currently Amended) A system for generating an intermediate representation of a register-based program code written for running on a programmable machine having a set of registers including at least one variable size register which is addressable by ~~said~~the program code in a plurality of different widths, the system comprising:

means for generating a plurality of register objects each representing a respective one of ~~said~~the registers as referenced by the program code, wherein ~~said~~the at least one variable ~~sized~~size register is represented by plural of ~~said~~the register objects with one ~~said~~the register object being provided for each different width of the variable ~~sized~~size register~~for holding variable values to be generated by the program code~~; and

means for generating a plurality of expression objects each representing a respective operator or operand according to ~~said~~the program code relating to ~~said~~the registers~~representing fixed values and/or relationships between said fixed values and said variable values according to said program code~~; and

means for forming a network of ~~said~~the register objects and the expression objects, wherein ~~with~~ each of ~~said~~the expression objects ~~being~~ is referenced by one or more of ~~said~~the register objects to which it relates either directly, or indirectly via references from other of ~~said~~the expression objects.

~~wherein at least one variable sized register is represented by plural register objects, one register object being provided for each possible size of the variably sized register.~~

11. (Currently Amended) A system for generating an intermediate representation of program code expressed in terms of the instruction set of a subject processor comprising ~~of~~ at least one ~~variably~~ variable sized register which is accessible by the program code in a plurality of different field widths, the system comprising:

means for generating a set of associated ~~abstract~~ register objects each representing a different field width of the variable size register~~representing the variably sized register~~;

means for writing, for each write operation of a certain field width to the variable sized register, to ~~an abstract~~the register object of the same width;

means for maintaining a record of which ~~abstract said~~the register objects contain valid data, the record being updated upon each write operation; and

means for determining from ~~said~~the record, for each read operation of a given field width, whether there is valid data in more than one of ~~said~~the ~~different sized abstract registers of the set~~associated set of register objects which must be combined to give the same effect as the same read operation performed upon the variable size register, and

(a) if it is determined that no combination is so required, reading directly from the appropriate register object; or

(b) if it is determined that data from more than one register objects must be so combined, combining the contents of those register~~s~~ objects.

12. (Currently Amended) The method of claim 1, wherein each of a plurality of ~~said~~ variable~~y~~ sized registers referenced by the program code is represented by a separately addressable subset~~s~~ of ~~said~~the register objects.

13. (Currently Amended) The method of claim 12, wherein each of ~~said~~the separately addressable subsets of the register objects concurrently represent the same respective variable~~y~~ sized register.

14. (Cancelled)

15. (Currently Amended) The method of claim 8, wherein ~~said~~each of a plurality of variable size registers referenced by the program code ~~variably sized register~~ is represented by a separately addressable subset~~s~~ of ~~abstract said~~the register objects.

16. (Currently Amended) The method of claim 15, wherein each of ~~said~~the separately addressable subsets of the ~~abstract~~ register objects concurrently represent the same respective variable~~y~~ size~~d~~ register.

17. (Cancelled)

18. (Currently Amended) The method of claim 10, wherein each of a plurality of variable size registers referenced by the program code ~~said variably sized register~~ is represented by a separately addressable subset~~s~~ of ~~said~~the register objects.

19. (Currently Amended) The method of claim 18, wherein each of ~~said~~the separately addressable subsets of the register objects concurrently represent the same respective variabl~~y~~e size~~d~~ register.

20. (Cancelled)

21. (New) An emulation method of convert subject code into target code, comprising the computer-implemented steps of:

(a) receiving the subject code expressed in an instruction set of a subject processor having at least one variable size register, wherein the variable size register is accessible by the subject code in a plurality of different width sub-fields overlaying part or all of a full width of the variable size register, and wherein the subject code includes at least write operations and read operations with respect to the variable size register;

(b) generating an intermediate representation from the subject code, including performing the steps of:

generating a set of associated register objects each representing a different width sub-field of the variable size register;

representing each write operation in the subject code of a certain sub-field width to the variable size register, as a write operation to the register object of the same width and maintaining a record of which the register objects contain valid data, the record being updated upon each such write operation; and

representing each read operation in the subject code of a certain sub-field width from the variable size register, as a read operation from one or more of the register objects by determining from the record whether there is valid data in more than one of the associated set of register objects which must be combined to give the same effect as the same read operation performed upon the variable size register in the subject code, and

(i) if it is determined that no combination is so required, reading directly from the appropriate register object; or

(ii) if it is determined that data from more than one register objects must be so combined, combining the contents of those register objects; and

(c) converting the intermediate representation into target code expressed in an instruction set of a target processor, including allocating the determined register objects to registers of the target processor and generating target code instructions which write to and read from the allocated target registers, according to the write operations and read operations defined in the intermediate representation.